Robert SMYK[1], Maciej CZYŻAK[2]

Rozdział 1. NOWY ALGORYTM ROZSZERZANIA BAZY W ARYTMETYCE RESZTOWEJ

1. Wstęp

Resztowy system liczbowy (ang. residue number system, RNS) może stanowić alternatywę dla binarnego systemu wagowego przy realizacji wielu algorytmów w cyfrowym przetwarzaniu sygnałów i RSA [5]. W RNS operacje dodawania, odejmowania i mnożenia wykonywane są niezależnie na poszczególnych parach cyfr operandów. Powoduje to, że nie występują przeniesienia między poszczególnymi pozycjami liczby. Jednak użycie RNS wymaga konwersji liczby reprezentowanej zwykle w systemie z uzupełnieniem do dwóch (U2) do systemu resztowego, a następnie odwrotnej konwersji wyniku z systemu resztowego do systemu U2. Inne operacje, takie jak dzielenie czy określanie znaku liczby oraz rozszerzanie bazy, są stosunkowo trudne do realizacji w systemie resztowym. Stąd system RNS może być atrakcyjny dla rozwiązań, gdzie dominują operacje dodawania/odejmowania i mnożenia. Taka sytuacja występuje przy realizacji algorytmów w cyfrowym przetwarzaniu sygnałów, takich jak filtracja FIR (ang. Finite Impulse Response) czy obliczanie szybkiej transformaty Fouriera. W wyniku mnożenia w RNS możemy uzyskać nadmiar arytmetyczny, co objawi się błędnym rezultatem operacji. Aby zapobiec nadmiarowi, po każdym mnożeniu konieczne jest zwykle zastosowanie skalowania, tzn. dzielenia przez pewną liczbę K, która została zastosowana do przekształcenia zwykle ułamkowych współczynników algorytmu na liczby całkowite. Algorytmy skalowania przedstawiono w licznych pracach [1-4]. Algorytm skalowania generalnie zależy od tego, czy baza

[1] Politechnika Gdańska, Wydział Elektrotechniki i Automatyki, Katedra Automatyki, ul. G. Narutowicza 11/12, 80-233 Gdańsk, e-mail:robert.smyk@pg.edu.pl
[2] Akademia Nauk Stosowanych w Elblągu, Instytut Informatyki Stosowanej im. K. Brzeskiego, ul. Wojska Polskiego 1, 82-300 Elbląg, e-mail: m.czyzak@ans-elblag.pl

systemu resztowego obejmuje pewien zestaw potęg liczby dwa, czy też stanowi zestaw modułów o długości binarnej zwykle 5, 6 bitów.

W pracy przedstawiono nowy algorytm rozszerzania bazy oparty na chińskim twierdzeniu o resztach (ang. Chinese Remainder Theorem, CRT). Rozszerzanie bazy polega na tym, że dla danej bazy systemu i zbioru reszt względem modułów bazy dla danej liczby X należy znaleźć resztę dla modułu lub modułów niewchodzących w skład bazy. Rozszerzenie bazy jest ogólną operacją w systemie liczbowym. Jest ona jednak szczególnie przydatna w przypadku wspomnianego wcześniej skalowania w systemie resztowym, gdyż przy skalowaniu przez współczynnik skalowania K, będący iloczynem wybranych modułów bazy, reszty dla modułów niewchodzących w skład K mogą być łatwo wyznaczone przez mnożenie każdego z nich przez odwrotność multiplikatywną każdego z tych modułów względem K. Stąd efektywność algorytmu skalowania w tym przypadku w sposób istotny zależy od efektywności algorytmu rozszerzania bazy o moduły wchodzące w skład współczynnika skalowania K. Innym bardzo istotnym zastosowaniem jest użycie algorytmu rozszerzania bazy w przypadku realizacji szyfrowania RSA z użyciem systemu resztowego. W algorytmie RSA opartym na mnożeniu metodą Montgomery konieczne jest dwukrotne zastosowanie algorytmu rozszerzania bazy. W związku z tym, podobnie jak w przypadku algorytmu skalowania, również przy realizacji tego algorytmu efektywność algorytmu rozszerzania bazy istotnie może wpłynąć na wypadkową efektywność obliczeniową algorytmu RSA. W sekcji drugiej omówiono podstawy systemu RNS, w sekcji trzeciej przedstawiono nowy algorytm rozszerzania bazy, w sekcji czwartej przedstawiono przykład numeryczny, w sekcji piątej architekturę układu realizującego nowy algorytm rozszerzania bazy, a w sekcji szóstej omówiono złożoność uzyskanej realizacji układowej.

2. Resztowy system liczbowy

Resztowy system liczbowy RNS jest zdefiniowany przez bazę $B = \{m_1, m_2, \dots, m_n\}$, czyli przez zbiór zwykle wzajemnie pierwszych nieujemnych liczb zwanych modułami. Zakres liczbowy M systemu resztowego zdefiniowany jest jako $M = \prod_{i=1}^{n} m_i$. M oznacza ilość liczb całkowitych, które mogą być reprezentowane przy użyciu danej bazy. Jeśli moduły są wzajemnie pierwsze, tzn. $gcd\left(m_j, m_k\right) = 1$ dla $j \neq k$, $j, k = 1,2, \dots, n$, to każda liczba całkowita X z zakresu $[0, M-1]$ może być reprezentowana przez wektor n liczb całkowitych $(x_1, x_2, \dots, x_n)$, gdzie $x_i = |X|_{m_j}$

rozumiane są jako reszty jednoznacznie reprezentujące w systemie RNS liczbę X. Operacje arytmetyczne dodawania, odejmowania i mnożenia w systemie RNS są definiowane jako

$$(x_1, x_2, \dots, x_n) \oplus (y_1, y_2, \dots, y_n) = (z_1, z_2, \dots, z_n), \tag{1}$$

gdzie $z_i = |x_i \oplus y_i|_{m_i}$, $i = 1,2, \dots, n$, a symbol $\oplus$ oznacza operację dodawania, odejmowania lub mnożenia przeprowadzaną w małych pierścieniach $R(m_i)$, $i = 1,2, \dots, n$. Warunek wzajemnej pierwszości modułów zapewnia, że odwzorowanie między pierścieniem modulo M a sumą pierścieni $R(m_i)$, $i = 1,2, \dots, n$ jest izomorfizmem. Odwzorowanie takie może być zrealizowane przy użyciu chińskiego twierdzenia o resztach lub konwersji z mieszanymi podstawami (ang. Mixed Radix Conversion, MRC) [1]. CRT ma następującą postać

$$X = \left| \sum_{i=1}^{n} M_i \, |M_i^{-1}|_{m_i} x_i \right|_M, \tag{2}$$

gdzie $M_i = \frac{M}{m_i}$, a $|M_i^{-1}|_{m_i}$ jest inwersją multiplikatywną M_i modulo m_i.

3. Nowy algorytm rozszerzania bazy

Algorytm ten polega na efektywnym użyciu CRT (2), które można też przedstawić jako

$$X = \left| \sum_{j=1}^{n} X_j \right|_M = \sum_{j=1}^{n} X_j - r \cdot M \, . \tag{3}$$

Rozszerzanie bazy o dodatkowy moduł m_{n+1} wymaga obliczenia reszty względem m_{n+1} dla danej liczby X reprezentowanej w systemie resztowym przy użyciu bazy $B = \{m_1, m_2, \dots, m_n\}$, zgodnie z zależnością $| X |_{m_{n+1}} = \left| \left| \sum_{j=1}^{n} X_j \right|_{\mathrm{M}} \right|_{m_{n+1}}$. Wymaga to jednak redukcji modulo M, która jest złożona obliczeniowo dla dużych M, sięgających 40 bitów w przypadku realizacji niektórych algorytmów CPS. Operacji tej można uniknąć, stosując CRT w postaci (3), prawa strona wzoru, i następnie redukcji modulo m_{n+1} jak poniżej

$$| X |_{m_{n+1}} = \left| \sum_{j=1}^{n} X_j \; - rM \; \right|_{m_{n+1}}, \tag{4}$$

gdzie X_j to projekcje ortogonalne. Powyższe równanie można przedstawić w zmodyfikowanych formach jako

$$|X|_{m_{n+1}} = \left| \left| \sum_{j=1}^{n} X_j \right|_{m_{n+1}} + |-rM|_{m_{n+1}} \right|_{m_{n+1}} \tag{5}$$

lub

$$|X|_{m_{n+1}} = \left| \left| \sum_{j=1}^{n} |X_j|_{m_{n+1}} \right|_{m_{n+1}} + |-rM|_{m_{n+1}} \right|_{m_{n+1}}. \tag{6}$$

Efektywność danego algorytmu rozszerzania bazy istotnie zależy od skutecznego sposobu obliczania współczynnika nadmiaru r w CRT. Zależność (3) można dalej zapisać, dzieląc stronami przez M w postaci

$$\sum_{j=1}^{n} \frac{X_j}{M} = \frac{X}{M} + r \tag{7}$$

Zakładając, że r jest dodatnią liczbą całkowitą oraz ułamek X/M jest liczbą z zakresu $[0,1)$, uzyskujemy

$$r = \sum_{j=1}^{n} \frac{X_j}{M} - \frac{X}{M} = \sum_{j=1}^{n} \frac{|x_j N_j|_{m_j} M_j}{M} - \frac{X}{M}, \tag{8}$$

gdzie $N_j = M_j^{-1}$ oznacza inwersję multiplikatywną M_j modulo m_j . Biorąc pod uwagę, że

$$m_j = \frac{\prod_{i=1, i\neq j}^{n} m_i}{M}, \; i = 1, \ldots, n, \tag{9}$$

otrzymujemy r w postaci

$$r = \sum_{j=1}^{n} \frac{|x_j N_j|_{m_j}}{m_j} - \frac{X}{M}. \tag{10}$$

Po przekształceniu (10) otrzymujemy

$$\frac{X}{M} = \sum_{j=1}^{n} \frac{|x_j N_j|_{m_j}}{m_j} - r. \tag{11}$$

Biorąc część całkowitą z obu stron, uzyskujemy

$$r = \left\lfloor \sum_{j=1}^{n} \frac{|x_j N_j|_{m_j}}{m_j} \right\rfloor. \tag{12}$$

Ułamki w (12) mogą mieć nieskończone reprezentacje binarne, w związku z tym ułamek powinien być zaokrąglony do ustalonej długości, co daje błąd $e_j, j = 1,2, \ldots, n$ dla każdego z ułamków

$$e_j = \frac{|x_j N_j|_{m_j}}{m_j} - \left\{\frac{|x_j N_j|_{m_j}}{m_j}\right\}, j = 1,2,\dots,n. \quad (13)$$

Przy obliczaniu r konieczne jest takie obcięcie lub zaokrąglenie poszczególnych ułamków, aby można było otrzymać wartość r taką jak z (12). Przyjmując zaokrąglenie w dół oraz to, że sumaryczny błąd ma spełnić warunek $e < 0.5$, mamy

$$e = \sum_{j=1}^{n}\left(\frac{|x_j N_j|_{m_j}}{m_j} - \left\lfloor\frac{|x_j N_j|_{m_j}}{m_j}\right\rfloor\right) < 0.5, \quad (14)$$

a dla każdego z ${\epsilon'}_j$ musi być spełniona poniższa zależność

$${\epsilon'}_j = \frac{|x_j N_j|_{m_j}}{m_j} - \left\lfloor\frac{|x_j N_j|_{m_j}}{m_j}\right\rfloor < \frac{1}{2n}. \quad (15)$$

Przy spełnieniu (15) otrzymujemy przybliżenie r w postaci

$$r' = \sum_{i=1}^{n}\left\lfloor\frac{|x_j N_j|_{m_j}}{m_j}\right\rfloor \quad (16)$$

i r' będzie należeć do przedziału $r' \in (r - 0.5, r)$. W związku z tym wartość indeksu nadmiaru można otrzymać jako

$$r'' = r = \lfloor r' \rfloor + 1\ . \quad (17)$$

Obliczony współczynnik nadmiaru r'' może być dalej użyty w formule (6) rozszerzania bazy.

Przykład 1. Obliczenie indeksu nadmiaru i jego użycie do rozszerzenia bazy

Współczynnik nadmiaru zostanie wyznaczony dwoma sposobami, bezpośrednio oraz z wykorzystaniem proponowanego w niniejszej pracy algorytmu. Następnie obliczone r'' zostanie wykorzystane do obliczenia nowej reszty przy użyciu algorytmu rozszerzania bazy.

Dokładną wartość współczynnika nadmiaru dla $X = 10000001$ można wyznaczyć z zastosowaniem

$$r = \frac{\sum_{j=1}^{n} X_i - X}{M}\ . \quad (18)$$

Przyjmujemy bazę RNS $B = \{32, 29, 27, 25, 23, 19, 17\}$ dla $n = 7$. Zakres liczbowy jest równy $M = 4653525600$. Zbiór reszt dla powyższych danych jest

następujący $x \leftrightarrow (1, 21, 18, 11, 1, 15, 16, 6)$. Projekcje ortogonalne (4) dla danego X mają wartości

$$X_1 = 3926412225,$$
$$X_2 = 3048861600,$$
$$X_3 = 1895880800,$$
$$X_4 = 4467384576,$$
$$X_5 = 4046544000,$$
$$X_6 = 3428913600,$$
$$X_7 = 2463631200.$$

Suma projekcji wynosi $\sum_{j=1}^{n} X_j = 23277628001$. Na podstawie (18) uzyskujemy $r = 5$.

Następnie obliczona zostanie wartość przybliżona r'' przy użyciu proponowanej metody. Ułamki $p_i = \frac{|X_j N_j|_{m_j}}{m_j}$ mają następujące wartości

$$p_1 = 0{,}84375,$$
$$p_2 = 0{,}655172413793103,$$
$$p_3 = 0{,}555555555,$$
$$p_4 = 0{,}96,$$
$$p_5 = 0{,}869565217391304,$$
$$p_6 = 0{,}736842105263157,$$
$$p_7 = 0{,}529411764705882.$$

Po zsumowaniu p_i uzyskujemy wartość $r = 5{,}002$, co można wyjaśnić błędem nadmiaru w arytmetyce zmiennoprzecinkowej. Po zaokrągleniu do 4 bitów zgodnie z (15) uzyskujemy następujące wartości $\tilde{p}_i$

$$\tilde{p}_1 = 0{,}8125,$$
$$\tilde{p}_2 = 0{,}625,$$
$$\tilde{p}_3 = 0{,}5,$$
$$\tilde{p}_4 = 0{,}9375,$$
$$\tilde{p}_5 = 0{,}8125,$$
$$\tilde{p}_6 = 0{,}6875,$$
$$\tilde{p}_7 = 0{,}5.$$

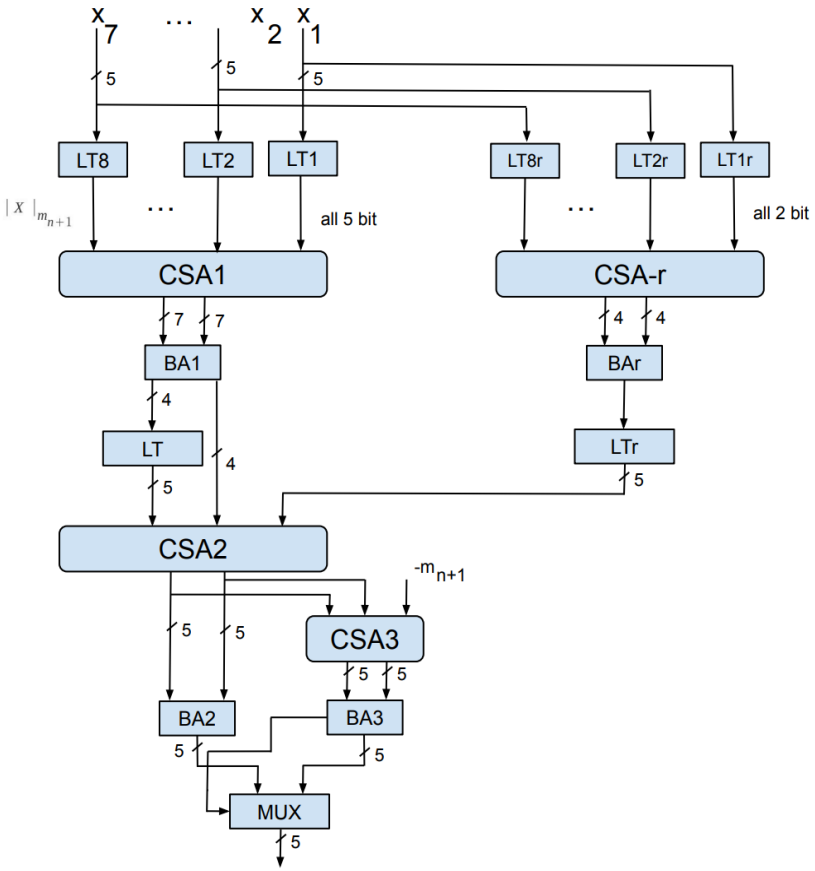
Finalnie otrzymujemy sumę ułamków obciętych jako $r' = 4{,}875$, co po zaokrągleniu daje $r'' = \lfloor 4{,}875 \rfloor + 1 = 5$.

Dana jest baza $B = \{32, 29, 27, 25, 23, 19, 17\}$. Baza ta będzie rozszerzona o moduł $m_8 = 31$ dla liczby $X = 10000001$. Reszta obliczona bezpośrednio ma wartość $|X|_{m_8} = 21$. Rozszerzenie bazy zostanie zrealizowane na podstawie wzoru (6). Projekcje ortogonalne mają takie same wartości jak wyżej. Wykorzystujemy wyznaczoną wcześniej wartość współczynnika nadmiaru w CRT, $r = 5$. Wartość reszty dla modułu, o który rozszerzamy bazę, obliczono bezpośrednio $x_8 = |X|_{m_8} = 21$. Wartość reszty wyznaczonej przy użyciu (6) wynosi $|X|_{31} = \big|\,|\,|23277628001|_{31}|_{m_{n+1}} + |-5 \cdot 4653525600|_{31}\big|_{31} = 21$.

4. Architektura układu rozszerzania bazy

Przedstawiona na rys. 1 architektura zaprojektowanego układu rozszerzania bazy (ang. base extender, BE) realizuje rozszerzenie bazy o jeden moduł na podstawie zależności (6). Układ składa się z dwóch kanałów. W kanale pierwszym obliczana jest suma modulo m_{n+1} projekcji ortogonalnych zredukowanych uprzednio modulo m_{n+1}. W kanale drugim obliczany jest współczynnik nadmiaru *r* w CRT. W kanale pierwszym $\big|X_j\big|_{m_{n+1}}$, $j = 1,2,\ldots,7$ obliczane są z zastosowaniem niewielkich funkcji logicznych, które mogą być realizowane przy użyciu małych pamięci typu ROM o rozmiarze $\lceil log\ \ m_j \rceil \times \lceil log\ \ m_j \rceil$ adresowanych przez poszczególne reszty x_j, $j = 1,2,\ldots,8$. Operandy uzyskiwane na wyjściu poszczególnych pamięci są sumowane przy wykorzystaniu drzewa sumatorów *CSA*1. CSA (ang. Carry-Save Adder) jest wielooperandowym drzewem sumatorów o liczbie warstw podanej w [10, s. 104], składającym się z *n*-2 sumatorów CSA dla *n*-operandów. Wektory wyjściowe drzewa CSA1, wektor sumy ***s*** oraz wektor przeniesienia ***c***, są sumowane przy użyciu sumatora binarnego *BA*1. Aby przeprowadzić redukcję modulo m_{n+1} sumy *S*, którą otrzymujemy na wyjściu *BA*1, binarny wektor sumy ***s*** jest dzielony na dwa krótsze wektory s_H i s_L w taki sposób, aby wektor s_L reprezentował liczbę $S_L$mniejszą od m_{n+1}. Liczba S_H, reprezentowana przez wektor s_H, jest redukowana modulo m_{n+1} z zastosowaniem odwzorowania $S_H \rightarrow |S_H|_{m_{n+1}}$ realizowanego przy użyciu małej pamięci typu ROM.

W kanale wyznaczającym *r* obliczane są wartości na podstawie zależności (17). Są one wyznaczane przy wykorzystaniu odwzorowania $x_j \to \left\lfloor \frac{|x_j N_j|_{m_j}}{m_j} \right\rfloor$. Następnie realizowane jest sumowanie przy użyciu drzewa CSA-r. Wektory wyjściowe reprezentujące sumę i przeniesienie są sumowane w BAr. W dalszej kolejności w LTr obliczana jest $s_{BAr} \to |S_{BAr}|_{m_{n+1}}$. W następnym kroku wykorzystywany jest sumator CSA2 obliczający sumę wartości z kanału pierwszego i kanału drugiego. Na wyjściu CSA2 zachodzi relacja $S_{CSA2} + C_{CSA1} < 2 \cdot m_{n+1}$; w związku z powyższym do redukcji modulo m_{n+1} można zastosować dwuoperandowy sumator modulo m_{n+1}. Sumator CSA3 oblicza w kodzie U2 sumę $S_{CSA2} + C_{CSA2} - m_{n+1}$. W kolejnym stopniu wykonywane są równolegle sumowania $S_{CSA2} + S_{CSA3}$. Na podstawie wartości przeniesienia z bitu znaku sumatora BA3 wybierana jest przez multiplekser wartość wyjściowa. Jeśli wystąpiło przeniesienie, wybierana jest wartość z sumatora BA3, jeżeli przeniesienie się nie pojawi, wybierana jest wartość z BA2. Przyjęto, że wszystkie sumatory BA w układzie są *n*-bitowymi sumatorami szeregowymi o strukturze RCA (ang. ripple carry adder).



Rys. 1. Architektura układu rozszerzania bazy dla $m_8 = 31$ i $B = \{32,29,27,25,23,19,17\}$

5. Analiza złożoności sprzętowej i opóźnień BE

Poniżej przeanalizujemy złożoność sprzętową i czasową zaprojektowanego układu rozszerzania bazy BE. Złożoność A_{BE} można wyrazić jako złożoność głównego kanału A_{BEm} oraz złożoność dodatkowego kanału A_{BEr} do obliczania współczynnika nadmiaru *r*.

$$A_{BEm} = n \cdot A_{LT(2^6\times 6)} + A_{CSA(n-2,a)} + A_{BA(2^{a-2},a)} +$$

$$A_{CSA(1,a)} + A_{CSA(1,a+2)} + A_{BA2(a)} + A_{BA3(a+2)}, \quad (19)$$

gdzie przyjmujemy, że $a = \lceil log \ \ m \rceil$, $A_{CSA(n-2,a)}$ oznacza złożoność *n-2* operandowego sumatora CSA sumującego operandy *a*-bitowe; podstawiając do (19) złożoności poszczególnych bloków architektury, otrzymujemy

$$A_{BEm} = nA_{LT}(2^a \times a) + (n-2)aA_{FA} + (a+2)A_{FA} +$$

$$A_{LT}(2^{a-2} \times a) + aA_{FA} + (a+2)A_{FA} + a0.6A_{FA}. \quad (20)$$

Po uzupełnieniu wartości zgodnie z projektem układu przedstawionym w poprzednim punkcie uzyskujemy

$$A_{BEm} = 8 \cdot 7{,}2A_{FA} + 7A_{FA} + 5A_{FA} + 5A_{FA} + \quad (21)$$

$$7A_{FA} + 5A_{FA} + 7A_{FA} + 0{,}6 \cdot 5A_{FA} = 96{,}2A_{FA}.$$

Złożoność sprzętową kanału do obliczania *r* można przedstawić następująco

$$A_{BEr} = nA_{LT(2^a\times(a-3))} + A_{BA(5)} + A_{LT(2^{a-2}\times a)}, \quad (22)$$

Gdzie

$$A_{LT(2^5\times 2)} = 64 \cdot 2 = 128 \sim 4A_{FA},$$

$$A_{LT(2^3\times 2)} = 8 \cdot 5 \cdot 2 = 80 \sim 3A_{FA},$$

$$A_{BA(5)} = 5A_{FA}.$$

Po podstawieniu uzyskujemy $A_{BEr} = 4A_{FA} + 3A_{FA} + 5A_{FA} = 12A_{FA}$.

Całkowita złożoność BE wynosi

$$A_{BE} = A_{BEm} + A_{BEr} = 96{,}2A_{FA} + 12A_{FA} = 108{,}2A_{FA},$$

co w przeliczeniu na tranzystory daje ~3462 tranzystory.

Całkowite opóźnienie BE można wyznaczyć jako

$$t_{BE} = t_{LT(2^a\times a)} + \theta(n)t_{FA} + t_{BA1} + t_{LT(2^{a-1}\times q)} + \quad (23)$$

$$t_{CSA2(1,a)} + t_{CSA3(1,a)} + t_{BA3(a+2)} + t_{MUX(2,1)}.$$

Dla $a = 5$ otrzymujemy następujące wartości poszczególnych opóźnień

$$t_{LT(2^5\times5)} = 7{,}2t_{FA},$$

$$t_{CSA(8,5)} = \theta(n)t_{FA},$$

$$t_{BA1} = 7t_{FA},$$

$$t_{LT(2^4\times5)} = 7{,}2t_{FA},$$

$$t_{CSA2(1,5)} = t_{FA},$$

$$t_{CSA3(1,5)} = t_{FA},$$

$$t_{BA3(7)} = 5t_{FA},$$

$$t_{MUX(2,1)} = 0{,}7t_{FA}.$$

Opóźnienie pamięci ROM zostało oszacowane na podstawie danych z biblioteki Samsung SDK 150 [9]. Liczbę warstw w drzewie CSA n-operandowym oznaczono przez $\theta(n)$ [10], co dla 7 operandów daje opóźnienie $4t_{FA}$. Opóźnienie sumatora pełnego przyjmujemy $t_{FA} = 0{,}269\,ns$ [9]. Założono, że opóźnienie każdego z sumatorów BA jest równe $l \cdot t_{FA}$, gdzie l to długość binarna operandów. W zaprezentowanej architekturze BE użyto BA o długości 4, 5 i 7 bitów. Uwzględniając powyższe, uzyskujemy opóźnienie układu rozszerzania bazy t_{BE} = 8,9 ns.

6. Porównanie

Dla wcześniej wymienionej bazy systemu RNS zaprojektowano układ według wcześniej wspomnianej metody Shenoya Kumaresana (SK), która jest uważana za najbardziej efektywną metodę rozszerzania bazy. Algorytm SK oparty jest na zastosowaniu dodatkowego modułu redundacyjnego, który pozwala na wyznaczenie współczynnika nadmiaru w CRT. Wadą tej metody jest konieczność użycia w całym układzie pracującym w RNS redundancyjnego kanału, co zwiększa złożoność sprzętową układu. Ponadto rozwiązanie SK wymaga zastosowania dużych pamięci ROM, co z jednej strony zwiększa opóźnienie BE, a z drugiej ogranicza w istotny sposób maksymalną częstotliwość potokowania. Układ SK wymaga pamięci ROM o binarnym adresie równym sumie długości binarnych modułów, przy założeniu że moduły są tej samej długości binarnej. Finalnie złożoność układu realizującego algorytm SK wynosi $A_{BE,SK}$ = 103312 tranzystorów.

7. Podsumowanie

W pracy przedstawiono nową metodę rozszerzania bazy opartą na CRT. Jednym z najbardziej istotnych elementów przy wykorzystaniu tego podejścia jest efektywne obliczenie współczynnika nadmiaru w formule CRT. Sposób obliczenia tego współczynnika ma bardzo istotny wpływ na złożoność realizacji układowej BE. Opisano nowy sposób obliczania tego współczynnika, umożliwiający wielokrotną redukcję złożoności sprzętowej kanału obliczającego współczynnik nadmiaru. Układ BE zawiera tylko małe pamięci typu ROM, które w razie potrzeby mogą być zastąpione odpowiednimi funkcjami logicznymi, co w efekcie daje możliwość potokowania na poziomie przerzutnika typu zatrzask. W związku z tym częstotliwość potokowania zależy tylko od opóźnienia pełnego sumatora i przerzutnika typu zatrzask. Układ BE nie wymaga operacji na dużych liczbach, obliczenia w systemie resztowym odbywają się tylko przy użyciu krótkich operandów binarnych.

Bibliografia

1. Szabo N.S., Tanaka R.I.: Residue arithmetic and its applications to computer technology. New York: McGraw-Hill, 1967.
2. Ulman Z., Czyżak M., Zurada J.: Effective RNS scaling algorithm with the Chinese remainder theorem decomposition. Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, vol. 2, May 1993, pp. 528-531.
3. Kong Y., Phillips B.: Fast scaling in the residue number system. IEEE Transactions on Very Large Scale Integration Systems (VLSI), vol. 17, no. 3, 2009, pp. 443-447.
4. Chang C.H., Low J.Y.S.: Simple, fast, and exact RNS scaler for the three-moduli set $\{2^n - 1, 2^n, 2^n + 1\}$. IEEE Transactions on Circuits and Systems I: -Regular Papers, vol. 58, no. 11, November 2011, pp. 2686-2697.
5. Garcia A., Lloris A.: A look-up scheme for scaling in the RNS. IEEE Transactions on computers, vol. 48, no. 7, July 1999, pp. 748-751.
6. Bajard J.-C., Imbert L.: A full RNS implementation of RSA. IEEE Transactions on computers, vol. 53, no. 6, June 2004, pp. 769-774.
7. Posch K.C., Posch R.: Modulo reduction in residue number systems. IEEE Transactions on Parallel and Distributed Systems, vol. 6, no. 5, May 1995, pp. 449-454.

8. Shenoy A.P., Kumaresan R.: Fast base extension using a redundant modulus in RNS. IEEE Transactions on Computers, vol. 38, no. 2, Feb. 1989, pp. 292-297.
9. Biblioteka Samsung STDH 150
10. Hwang K.: Computer arithmetic. Principles, Architecture and Design, John Wiley & Sons, 1979.

Streszczenie

Przedstawiono nowy algorytm rozszerzania bazy w resztowym systemie liczbowym bez użycia nadmiarowego modułu. Rozszerzanie bazy jest kluczową operacją w wielu zastosowaniach, gdzie używany jest resztowy system liczbowy, takich jak cyfrowe przetwarzanie sygnałów, jak też implementacja systemu szyfrowania algorytmem Rivesta-Shamira-Adlemana (RSA). Ortogonalne projekcje występujące w chińskim twierdzeniu o resztach dla modułu, o który jest rozszerzana baza systemu resztowego, przechowywane są w niewielkich pamięciach typu ROM (ang. Read-Only Memory) w formie resztowej. Projekcje te są sumowane w sumatorze binarnym i otrzymana suma jest redukowana dla modułu, o który rozszerzana jest baza. Metoda wykorzystuje nowy i efektywny algorytm obliczania współczynnika nadmiaru w chińskim twierdzeniu o resztach. Umożliwia ona wielokrotną redukcję złożoności sprzętowej.

Słowa kluczowe: arytmetyka resztowa, rozszerzanie bazy.